Homeostatic Fault Tolerance in Spiking Neural Networks: A Dynamic Hardware Perspective

Homeostatic Fault Tolerance in Spiking Neural Networks: A Dynamic Hardware Perspective

*Anju P. Johnson *Member, IEEE*, **Junxiu Liu, *Member, IEEE*, *Alan G. Millard, **Shvan Karim, *Andy M. Tyrrell *Senior Member, IEEE*, **Jim Harkin *Member, IEEE*, *Jon Timmis *Senior Member, IEEE* **Liam McDaid and *David M. Halliday

***Abstract*—Fault tolerance is a remarkable feature of biological systems and their self-repair capability influence modern electronic systems. In this work, we propose a novel plastic neural network model which establishes homeostasis in a spiking neural network. Combined with this plasticity and the inspiration from inhibitory interneurons, we develop a fault-resilient robotic controller implemented on an FPGA establishing obstacle avoidance task. We demonstrate the proposed methodology on a spiking neural network implemented on Xilinx Artix-7 FPGA. The system is able to maintain stable firing (tolerance $\pm 10\%$) with a loss of up to 75% of the original synaptic inputs to a neuron. Our repair mechanism has minimal hardware overhead with a tuning circuit (repair unit) which consumes only 3 slices/neuron for implementing a threshold voltage based homeostatic fault-tolerant unit. The overall architecture has a minimal impact on power consumption and therefore supports scalable implementations. This work opens a novel way of implementing the behavior of natural fault tolerant system in hardware establishing homeostatic self-repair behavior.**

Keywords*—Self-Repair, Homeostasis, Fault Tolerance, FPGA, Dynamic Partial Reconfiguration, Bio-inspired Engineering, Mixed-Mode Clock Manager, Phase Locked loop.

I. Introduction

Bio-inspired solutions are playing an important role in solving real-world engineering problems [1]. Healing is a remarkable feature of the brain which leads to the restoration of cognitive function following stroke or injury. Additionally, the human brain is continuously undergoing modifications to adapt to changes in its environment [2]. This follows from the fact that the brain is capable of assessing its own activity levels and performing adjustments to maintain a stable operation. This work is based on the inspiration derived from robust biological systems, which can detect and correct errors. Homeostasis is a property of a system to maintain relatively stable equilibrium when subjected to continuous change. The mechanisms that monitor excitation and maintain the functional properties of neurons are by definition homeostatic [3]. Homeostasis can be established in two ways: one in which neurons alter their intrinsic electrical properties and the other by modifying synaptic properties to maintain a target level of electrical activity. Recent work has shown that destabilizing influences (e.g. decaying synapses) are counterbalanced by homeostatic plasticity mechanisms that act to stabilize neural and circuit activity [3], [4]. Synapses connecting between neurons are the most vital information processing unit in an Artificial Neural Network (ANN) system [5]. Any faulty behavior in synapses affects the performance of the entire system. Hence this work considers hardware failures targeting synapses of a neuromorphic system implemented on an FPGA. Specifically, we consider a scenario in which a neuron tries to establish homeostasis in the presence of synaptic failures by altering the intrinsic electrical properties. The inbuilt nature of the system also helps in addressing problems of sensor failures.

There are many sources of errors in electronic systems including single and multiple event upsets, aging faults, power supply fluctuations, thermal instabilities, metal migration, hot carrier injection, and oxide breakdowns which are becoming very common in electronics systems [6]. These may lead to soft and/or hard errors in the systems. As feature size and operating conditions are scaled down, the sensitivity of electronic devices to radiation at ground levels has also increased [7].

Spiking neurons are core components of many computational models of the brain that aim to improve understanding of brain function and hence its fault tolerance is of utmost importance. So, in this work, we propose a novel way to achieve fault tolerance in Spiking Neural Network (SNN) implementation on FPGA. In SNNs, communication, and computation happen by an exchange of spatiotemporal patterns encoded as spikes as in biological neurons. Various researchers have demonstrated fault tolerance in SNN hardware implementations [8]–[10]. Compared to these works, the work proposed in this paper demonstrates higher fault tolerance and the methodology is feasible in the presence of at least one healthy synapse.

In biological systems, independent units perform computation in parallel. For real world applications, this parallelism can be exploited to perform tasks orders of magnitude faster than in software and hence we consider hardware implementation of SNNs. In this work, we use FPGAs to demonstrate fault tolerance inherent in SNNs, because they combine computing capability, logic resources and memory capacity in a single device. The potential of fault tolerance in FPGA-based SNN networks has been identified in some of the recent works [11], [12]. FPGA allows neural networks to be evolved on hardware and new topologies/networks executed faster [13]. In this paper, we focus on neural networks implemented on Static Ran-

The authors are with the **Intelligent Systems & Nano-science Research Group*, Department of Electronic Engineering, University of York, Heslington, York YO10 5DD, UK and ***Intelligent Systems Research Centre*, School of Computing and Intelligent Systems, Ulster University, Derry BT48 7JL, UK. Email:{anju.johnson, david.halliday, alan.millard, andy.tyrrell, jon.timmis}@york.ac.uk, {j.liu1@, haji_karim-s@email., jg.harkin@, lj.mcdaid@}ulster.ac.uk,

dom Access Memory (SRAM)-based FPGA since it is the most commonly used reconfigurable platform. SRAM-based FPGAs are particularly prone to *Single Event Upsets* (SEUs) [14]. This creates an issue for dependability for safety critical applications. Emerging technologies for hardware implementations such as memristor technology in FPGA like devices [15] are a promising solution for future implementations. Memristors allow mapping of a high-density integrated circuit with clock speeds in the gigahertz range. A single memristor can perform functions of multiple transistors, leading to the fabrication of a powerful computer. Memristor is an attractive alternative to SRAMs and flash storage. Memristor-based neural networks have been reported in some of the recent researches [16]–[18]. Logic circuits based on magnetic RAM (MRAM) can be an energy-efficient replacement for SRAM-based FPGAs [19]. They offer zero leakage and CMOS compatibilities.

In this work, we demonstrate homeostatic fault tolerance in SNN implemented on an FPGA using (a) neuronal threshold voltage adjustment, and (b) Dynamic Partial Reconfiguration (DPR) of neuronal clocking schemes. This work considers faults as a condition that results in a silent or near silent neuron caused by low transmission probability (PR) of a synapse. A neuron is a representation of a node in a system. Near silent neuron presents a weak node in the system. The issues might be hardware failures in the system. The proposed work is a solution for all kinds of faults leading to failures in the interconnections of the node (neuron). These may be sensor failures, SEUs, stuck-at fault, interconnect fracture, noise, etc.

Faults in synapses that lead to reduced transmission probability may be due to an external cause such as sensor failures or internal faults such as SEUs in synaptic connections. Repair is the ability of the system to restore firing rates. The idea we propose is to use variable threshold voltage adjustments for the neuron to handle low transmissions. DPR is an FPGA-specific technological advancement which aims at modifying the existing circuit mapped on the FPGA without needing to turn off the circuit functioning in other parts of the FPGA. Various works have demonstrated the possibility of fault tolerance in FPGAs via DPR [20], [21]. We use Dynamic clock alteration, a variant of classical DPR technique to establish the homeostatic repair in the presence of faulty synapses. The proposed mechanisms modify the threshold voltage or clock rate for the faulty neurons to effectively restore the firing rate to the original value. Additionally, this work compares the two approaches.

In many mobile robot navigation applications, one of the primary tasks is to avoid obstacles. The work demonstrates the effectiveness of the proposed bio-inspired homeostasis of SNN implemented on FPGA in achieving a robust obstacle avoiding robotic task. This obstacle avoiding robotic controller is designed with the help of homeostatic fault tolerance combined with and synaptic excitatory and inhibitory plasticity. We provide a complete architecture establishing homeostasis in an FPGA-based robotic controller having movements in ‘Forward’, ‘Right’, ‘Left’ and ‘Reverse’ directions. We also propose an area-reduced model of the robotic car controller establishing the same task. The equivalence between the two models is demonstrated.

The rest of the paper is organized as follows. Section II describes the background and motivation. Section III discusses the basic building block of the bio-inspired architecture. Section IV presents the proposed idea of neuronal self-tuning for homeostatic regulation of firing rates. Section V describes the two architectures of the robotic controller establishing obstacle avoidance task. Section VI presents experimental results establishing the effectiveness of the proposed schemes. Finally, conclusions are derived in Section VII.

II. Background and Motivation

A Spiking Neural Network (SNN) is a typical bio-inspired neural network that performs information transfer based on discrete-time spikes. When an SNN neuron receives an input spike, its membrane potential increases slowly and gradually drops due to leakage of ion channels. When multiple input spikes are received in rapid succession, the membrane potential may increase and reach a specific value (threshold), and the neuron fires a spike. There are multiple possible levels of abstraction for SNN modeling. Hodgkin-Huxley model [22] is the most biologically realistic neuron model. Other models include Quadratic Integrate and Fire (QIF), Exponential Integrate and Fire (EIF) [23], Izhikevich [24], FitzHugh-Nagumo [25], Hindmarsh-Rose [26] and Integrate and Fire (IF) model [27]. The Leaky Integrate and Fire (LIF) model, a variation of IF model is a simplified model of a biological neuron widely used in neuromorphic computing. It represents a good trade-off between computational complexity and biological realization. We use a LIF [28] model for representing the neurons in the network. This representation of a LIF neuron is shown in Eq. (1).

$$\tau_{mem}\frac{dv}{dt} = -v(t) + R_{mem}\sum_{i=1}^{m} {I_{syn}}^{i}(t) \tag{1}$$

where I_{syn}, R_{mem}, V_{th}, τ_{mem}, v, and V_r are current injected by a synapse, membrane resistance, threshold voltage, time constant, membrane potential, and resting membrane potential respectively. Typical values are I_{syn} = $20nA$, R_{mem} = $1M\Omega$, V_{th} = $15mV$, τ_{mem} = $10ms$, V_r = $0v$. m represents the number of synapses associated with a neuron. On reaching the threshold voltage, the membrane potential is brought back and held at 0V following a nominal refractory period (Δ_{abs}=2 clock cycles). The expression is evaluated using Euler method of integration with a fixed time step of $\Delta t = 2^{-10}s$ (an approximation for $1ms$). Considering the above parameters and a constant input current, solution for the above equation turns to be:

$$v(t) = R_{mem} \times I[1 - exp(-\frac{t}{\tau_{mem}})] \tag{2}$$

The asymptotic value of the membrane potential is $R_{mem} \times I$. If this value is less than the spiking threshold, V_{th}, no spike can be generated. If $R_{mem} \times I > V_{th}$, the neuron fires. The typical value of membrane resistance falls in the range of $M\Omega$, hence we fixed this to be $1M\Omega$. Additionally, the threshold voltage is selected to be $15mV$, leading to the value of I_{syn}. The parameters are derived by considering references

[8], [29]. SNN projects the mechanism of the brain based on massively parallel arrays of neurons. Hence to mimic the brain-like functionality, it is advisable to implement the spiking neural network in hardware as it takes advantage of inherent parallelism and very high execution speed. In this work, the proposed methodology superimposed on an obstacle avoiding robotic controller is implemented on an FPGA.

Homeostatic plasticity is a mechanism which regulates average activity in neural networks. Activity levels in nervous system rely on many factors including various plasticity mechanisms, environmental variations, and developmental changes. Homeostatic plasticity is a biological process in neurons that serves to compensate for such disruptions. Researchers have distinguished the effects of homeostatic regulations in many species including neuronal cultures [30], [31] and organotypic cultures [32], [33]. These studies provide evidence that intrinsic properties of the system are subject to activity-dependent regulation that maintains an average electrical activity. Various models of intrinsic homeostatic plasticity have also been developed. It was proposed that neurons have a built-in sensor mechanism that monitors electrical activity and adjusts conductance densities to maintain a specific activity level [34]. In this work, we propose precisely, such an intrinsic homeostatic mechanism in hardware where a neuron monitors its input activities and switches between predetermined intrinsic parameters. To the best of our knowledge, this is the first attempt to establish intrinsic homeostasis in hardware using an SNN model.

Together with the motivation from the intrinsic homeostatic mechanism, the proposed hardware architecture draws inspiration from the behavior of interneurons. Interneurons are a vital component in a neuron system which plays an important role between sensory and motor neurons. This work specifically targets inhibitory interneurons. In a biological system, inhibitory neuron release neurotransmitters (glycine and GABA) that bind to the corresponding receptors in the postsynaptic neurons and triggers a negative change in the membrane potential. When a muscle spindle is stretched, the antagonist muscle group must be inhibited to prevent it from working against the resulting contraction of the homonymous muscle [35]. Inhibitory interneurons in the spinal cord aids in establishing this task. In this work, we derive the working of inhibitory interneurons for establishing inhibitions in directions requiring less activity for establishing obstacle avoidance task.

III. The Basic Building Block

The basic unit of a robotic controller architecture is shown in Figure. 1. The architecture consists of a neuron (N1), associated with a set of *n* excitatory synapses (EX) and *m* inhibitory synapse (IN) ($n = 60$ and $m = 40$ in our experiments). The excitatory synapses receive readings from the sensor corresponding to the distance from the obstacle (6bit data). The robotic field is a ($10X10$ meter) square arena (can be varied). This distance (10 meters) is divided into 60 steps as the robot moves. The distance is mapped in binary where each step of the robot movement corresponds to shifting a logic one to the distance vector. If the input to a synapse is one, a Poisson spike train is provided to the synapse. Otherwise, the Poisson spike input to the synapse is disabled. In our experiments, we used Poisson spike trains, which on an average, produce one spike per 4 clock cycles. If the synapse receives a spike from the Poisson generators, after a short delay, the synapse outputs a constant current of $20nA$ to the LIF neuron for a duration of one clock cycle. The inhibitory synapses receive input from other neurons in the system. If the inhibitory synapse receives a spike input from a neuron, after a short delay, the synapse outputs a constant current of $-20nA$ to the LIF neuron for a duration of two clock cycles. This difference in delay is added to incorporate the difference in time constants of excitatory and inhibitory synapses. Inhibitory synapses generally have a higher time constant [36]. In addition to the above units, two tuning units are provided in the system for establishing homeostasis and fault tolerance. The tuning unit (T-1(EX)) monitors the current injected from the excitatory synapses and modulates neuron intrinsic parameters similar to the intrinsic homeostatic mechanism in the biological system. Similarly, the tuning unit (T-2(IN)) sits between the inhibitory synapses and the neuron. More details of the proposed tuning units are described in section IV. One aspect of our model is that it operates at an accelerated biological time scale similar to that in [14].

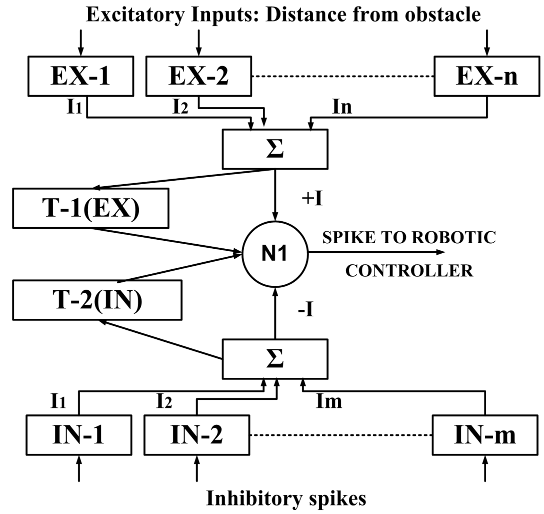

Fig. 1: **Basic Building Block of the Robotic Controller** Neuron N_1 receives n excitatory synaptic inputs representing distance from the obstacle. Additionally, m inhibitory synaptic inputs are also received from the inter neurons. Two tuning circuitries (T-1 (EX) and T-2 (IN)) establish homeostasis and fault tolerance.

IV. Neuron Network Incorporating Self-Tuning

In principle, a neuron could establish a constant firing rate through various mechanisms. This work achieves homeostatic regulation of firing rates using self-tuning of intrinsic parameters (threshold voltage and neuron clock frequency). The novelty lies in the following aspects:

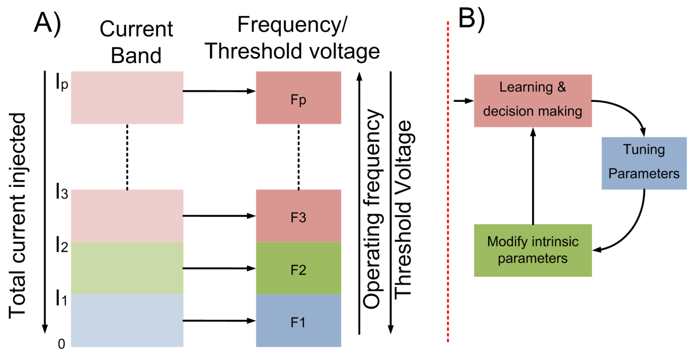

Fig. 2: **Illustration of proposed self-tuning methodology** (A) The maximum injected current falls at a time slot Δt under one of the current band $I_i - I_j$. The current falling in each current band are mapped to corresponding operating frequencies of the neural clock. As the maximum injected current falls in higher order bands, corresponding mapped threshold voltage/operating frequency of the neuron decreases. (B) The neuronal self-tuning is performed following three phases, namely, (1) monitoring the maximum current injected to the neuron and making a decision based on observed maximum current, (2) modeling of tuning parameters (intrinsic parameters), and (3) performing tuning.

1) **Variable threshold voltage of neuron:** The neurons in the system operate with different threshold voltages. For example, if the neuron has a reduction in spiking activity, the homeostatic regulatory mechanism comes into action, thereby lowering the threshold voltage of the neuron, causing the firing rate to increase.
2) **Dynamically tunable neurons:** Neurons with DPR-based tuning unit have the capability of self-tuning their operating frequencies. This is established using dynamic partial reconfiguration of *Phase Locked Loop* (PLL) module or the Mixed-Mode Clock Manager (MMCM) module in the FPGA. For example, if the neuron has a reduction in spiking activity, the homeostatic regulatory mechanism comes into action, thereby increasing the neuron clock frequency, which increases the firing rate.
3) **Variable operating frequency:** If *DPR-based tuning scheme* is employed, the various components of the system operate at different clock frequencies. For example, if the global clock frequency of the system is $20MHz$, the neurons in the system operate at a pre-specified frequency up to $20MHz$. In this work, the global clock to the system is designed using a PLL macro and the neuron clock generator is an MMCM module.

A. *The Proposed Methodology of Neuronal Self Tuning*

This work establishes self-tuning of neurons in two ways (a) using threshold voltage adjustment of the neuron, and (b) dynamic reconfiguration of neuron clock generator.

The operation of the proposed system can be summarized as follows: All synapses associated with a neuron inject a constant amount of current (I_{inj}: +I or -I) to the neuron. Based on the probabilistic nature of the synapse, the total current injected to the neuron varies with time. Considering a time slot for observation, the maximum current injected to the neuron remains constant in the absence of synaptic faults or obstacles. In the case of synaptic failures/obstacles, the maximum current injected to the neuron diminishes. All neurons in the system monitor the maximum current injected for a duration Δt. Based on this observation, the neurons decides whether or not to self-tune its intrinsic parameters. If the maximum current injected during a time slot of Δt_i is higher than or lower than the previous time slot Δt_{i-1}, the neuron generates necessary signals to the neuron control unit to modify the threshold voltage or clock frequency of the neuron. This allows the neuron to maintain a constant firing rate even if the total injected current reduces due to synaptic failures. Additionally, this helps to maintain a stable firing rate until the obstacle is $2.5m$ away from the robot, followed by a smooth reduction in firing rate (which is the more desirable behavior).

The choice of time slot Δt is decided based on two measures: (1) How fast is it required to establish a repair? If the neuron could determine the maximum current in a shorter duration a faster repair is possible. (2) Secondly, due to the random nature of synaptic fault and obstacles, the total current injected to a neuron at an observation slot may vary. This might trigger accidental tuning of neuron intrinsic parameters. This may not be an issue for the threshold voltage adjustment scheme, but is a concern for the DPR based tuning scheme and may cause unstable operations. Although no studies discuss the power consumed by an FPGA clock management unit during a DPR, it is widely accepted that DPR is a power consuming operation and should be avoided if unnecessary [37]. Considering low power applications, it is advisable to prevent unnecessary DPR and hence we provide a sufficient duration (2μs in our experiments) to monitor the maximum current injected to the neuron.

The self-repairing hardware paradigm presented in Figure. 2 shows three phases of the hardware cycle required to perform neuronal self-tuning. The first phase is the learning and decision-making phase. The neuron learns the maximum current injected into it. To illustrate the self-tuning concept we first consider the case where x out of 100 synapses associated with neuron N_1 are faulty. The maximum current that can flow to neuron N_1 at any time during the existence of a fault is $(100-x)I_{inj}$. The neuron monitors the total injected current to obtain a baseline measurement. Based on the maximum injected current, the neuron makes a decision whether or not to undergo a threshold voltage/operating frequency change. If the maximum injected current in slot Δt_i varies from that in slot Δt_{i-1}, a change in neuron intrinsic parameters is triggered. In the case of threshold voltage adjustment scheme, the neuron lowers the threshold voltage if the total injected current is decreased whereas, for frequency based tuning, operating frequency of neuron is increased. The details and range of tuning parameters for threshold voltage adjustment scheme are discussed in Section IV-C.

For DPR based clocking scheme, a frequency increase is

TABLE I: **Current bands to clock frequency mapping for neuronal self tuning: values designed empirically**

Percentage of synaptic Fault	Distance from obstacle (m)	Current injected I_{max}	Frequency (MHz)
$[0.00-3.33)\%$	$10.00-9.67$	$(60.I_{inj}-32.I_{inj})$	10
$[3.33-15)\%$	$9.67-8.50$	$(32.I_{inj}-29.I_{inj})$	12
$[15.00-26.67)\%$	$8.50-7.33$	$(29.I_{inj}-25.I_{inj})$	13
$[26.67-36.67)\%$	$7.33-6.33$	$(25.I_{inj}-21.I_{inj})$	14
$[36.67-46.67)\%$	$6.33-5.33$	$(21.I_{inj}-19.I_{inj})$	15
$[46.67-53.33)\%$	$5.33-4.67$	$(19.I_{inj}-14.I_{inj})$	16
$[53.33-100)\%$	$4.97-0.00$	$(14.I_{inj}-0)$	17
Note: In our experiments $I_{inj}=20nA$. Hence the band $60.I_{inj}-32.I_{inj}$ corresponds to current in the range $(1200-640)A$.			

TABLE II: **Current bands to threshold voltage mapping for neuronal self tuning: values designed empirically**

Percentage of synaptic Fault	Distance from obstacle (m)	Current injected I_{max}	Neuron V_{th} (mV)
$[0.00-3.33)\%$	$10.00-9.67$	$(60.I_{inj}-39.I_{inj})$	11.2304
$[3.33-15)\%$	$9.67-8.50$	$(39.I_{inj}-34.I_{inj})$	7.3242
$[15.00-26.67)\%$	$8.50-7.33$	$(34.I_{inj}-29.I_{inj})$	3.4179
$[26.67-36.67)\%$	$7.33-6.33$	$(29.I_{inj}-25.I_{inj})$	3.1738
$[36.67-46.67)\%$	$6.33-5.33$	$(25.I_{inj}-21.I_{inj})$	2.9296
$[46.67-53.33)\%$	$5.33-4.67$	$(21.I_{inj}-19.I_{inj})$	2.6855
$[53.33-100)\%$	$4.97-0.00$	$(19.I_{inj}-0)$	2.4414
Note: In our experiments $I_{inj}=20nA$. Hence the band $60.I_{inj}-39.I_{inj}$ corresponds to current in the range $(1200-780)A$.			

desired and the neuron formalizes the MMCM tuning parameters. The details and range of tuning parameters for neuron clock frequency adjustment are discussed in Section IV-B. The final phase is to perform threshold voltage adjustment or DPR. The neuron writes the new Vth value to the neuron threshold voltage input or DPR parameters on the *Dynamic Reconfiguration Port* (DRP) ports. This configures a new threshold voltage on the neuron or initiates a DPR at its associated clock management unit. Note: the proposed schemes of neuron tuning (threshold voltage or DPR based) works independently and only one or the other is used (not both).

B. Self-Tuning based on DPR of Neuronal Clocks

DPR in clock management tiles of the FPGA provides a way for generating custom clocks on the fly depending on the requirements of applications. The usual techniques to generate such custom clocks is to use clock generation circuitry such as the *Phase Locked Loop* (PLL) module or the Mixed-Mode Clock Manager (MMCM) module [38], [39]. The MMCM (or PLL) module is enabled in the *Clock Management Tiles* (CMTs) of the FPGA. This approach of DPR of clocking circuits is found to be useful in applications such as dynamic power management, software defined radios and random number generation [40]–[42]. We denote F_{CLKIN} the input clock signal to the MMCM and F_{CLKFX} the corresponding synthesized clock signal. We further define two major attributes of the MMCM module - $CLKFX_{MULTIPLY}$ attribute with value M and $CLKFX_{DIVIDE}$ an attribute with value D. The relation between the input and output clock signals is given by Eq. (3).

$$F_{CLKFX} = F_{CLKIN} \times \frac{M}{D} \tag{3}$$

The DPR capability of the FPGA allows modification of the M and D values during runtime to synthesize different clock frequencies. An MMCM controller generates the necessary control signals for the DRP ports of the MMCM module to write the target M and D values, which are passed on via the data input bus. The possible ranges for M and D are specific to the FPGA family used. F_{CLKIN} is designed to be $20MHz$ in our implementation. The tunable neurons in the system operate in the range $10MHz-20MHz$ and all other modules work at $20MHz$ clock frequency.

The total current injected to a neuron will fall into different current bands depending on the number of synaptic inputs. Based on the number of bands, the mapping from maximum injected current to the specific operating frequencies can be established. We illustrate the proposed idea by dividing the input current into seven bands. Based on the experimental observation, we have determined the required operating frequencies of the neuron in the presence of various fault percentages. This is depicted in Table I.

In reference to the degree of faults, the neuron increase/decrease its clock frequency and hence establishes a constant firing rate. Considering the modeling parameters and Eq. (3), we get frequencies of $10MHz$ to $17MHz$ for different current bands.

C. Self-Tuning based on Threshold Voltage Adjustment

Although DPR based clock frequency synthesis works for a wide range of values (7 series Xilinx FPGAs support fractional (non-integer) multiplication and division parameters for the clock synthesis) [38], [39], they are not suitable for large-scale implementation. Each 7 series Xilinx FPGA has up to 24 Clock management Tiles (CMTs), each consisting of one MMCM and one PLL module. Hence we have a total of 48 dynamically reconfigurable clock management units which impose a bottleneck on the number of tunable neurons. Hence, to overcome this issue we have come up with the idea of a tunable threshold voltage scheme in the neurons.

We define a standard threshold voltage (V_{sth}) of $15mV$ in our design. The relation between the standard threshold voltage and the instantaneous threshold voltage V_{th} of the neuron is given by Eq. (4).

$$V_{th} = V_{sth} \times R \tag{4}$$

where R is a rational number. The maximum injected current falls at a time slot Δt under one of the current band $I_i - -I_j$. Based on the current bands a new V_{th} is calculated during the tuning phase. This is provided to the neuron threshold input, thereby creating a time-varying value for the threshold voltage. This value is updated at each time slot in a 32-bit register holding the neuron threshold membrane potential. We illustrate the proposed idea by dividing the input current into seven bands. Based on experimental observation, we have determined the required threshold voltage of the neuron in the presence of various fault percentages. This is depicted in Table II.

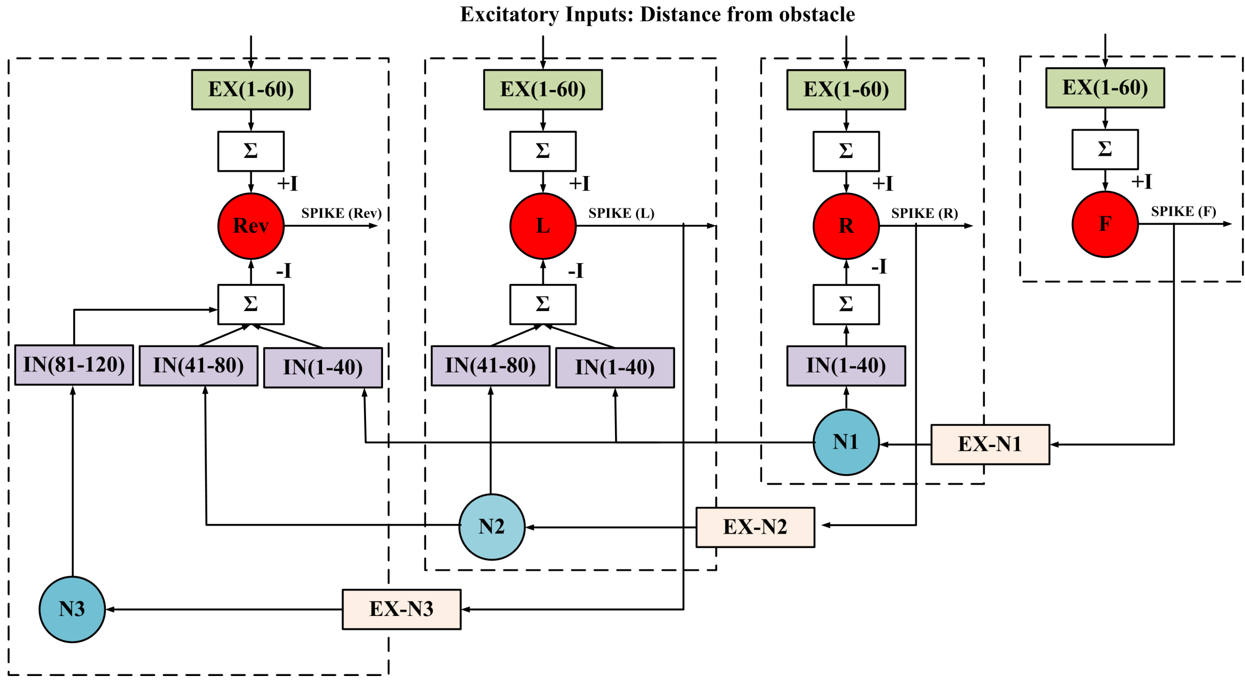

Fig. 3: **Complete circuitry of the robotic controller implemented on the FPGA** The neurons F,R,L and Rev corresponds to neurons driving motors for forward, right, left and reverse directions of the robotic car respectively. $N1$, $N2$ and $N3$ are interneuron. EX represents excitatory synapses and IN corresponds to inhibitory synapses. The controller priorities movements in direction in order of preference: $Forward > Right > Left > Reverse$. Inhibition acts to suppress movements in directions of lower priority. (Tuning blocks are not shown in figure.)

Fine tuning of the frequency or the voltage is not necessary for establishing the homeostatic behavior. The usage of a smooth function alters the threshold voltage or clock frequency at every small change of the input current to the neuron. This leads to unnecessary switching power. Additionally, for DPR based schemes, the smooth function would lead to greater number of DPR to happen and should be eliminated (DPR is treated as a power hungry operation). Fine-grained updates of neuron parameters would also cause delays in the system. Hence in our implementation, we divided the total injected currents into a few manageable bands. The band structure is derived empirically, to provide a smooth spiking behavior. Fundamentally, this enables the control of the level of power dissipation in hardware.

V. Architectures for Robotic Controller Establishing Obstacle Avoidance

The main aim of the proposed work is to implement fault tolerant behavior of SNN in hardware. Once this is established successfully, we apply the concept to a real-world task. The emphasis of the selected application was on the fault recovery aspect rather than establishing an obstacle avoidance task. Advantages are the inherent fine-grained (distributed) repair mechanism incorporated at level of individual synaptic inputs. The proposed homeostatic SNN system implemented on an FPGA performs an obstacle avoidance task. The architecture is tested with a range of obstacle conditions and faulty synapses. The homeostasis and fault tolerance is demonstrated in two architectures. The complete architecture incorporates the features of homeostasis and inhibitory interneurons to design a robotic controller paying attention to biological features. The reduced architecture is a simplification of the complete architecture by incorporating certain observations leading to a reduction in hardware overhead, power and delay features on the FPGA.

A. *Complete Architecture of the Robotic Controller Establishing Obstacle Avoidance*

Figure. 3 represents the complete architecture implemented on an FPGA establishing the obstacle avoidance task. The system consists of 4 motor neurons F (Forward), R (Right), L (Left) and Rev (Reverse) generating spike trains for motor movements in the forward, right, left and reverse directions respectively. Each of the motor neurons receives 60 excitatory inputs according to the distance from the obstacle. All of the excitatory inputs to the neuron being low represent a close encounter with the obstacle and all of them being high indicates no obstacle in the path for a distance of $10 meters$. In the former case, the neuron spikes at a rate of $250 spikes/10us$, and in the later case, we have a spike rate of 0. The speed of the wheel is directly proportional to the activity of the motor neuron, allowing movement in the absence of an obstacle.

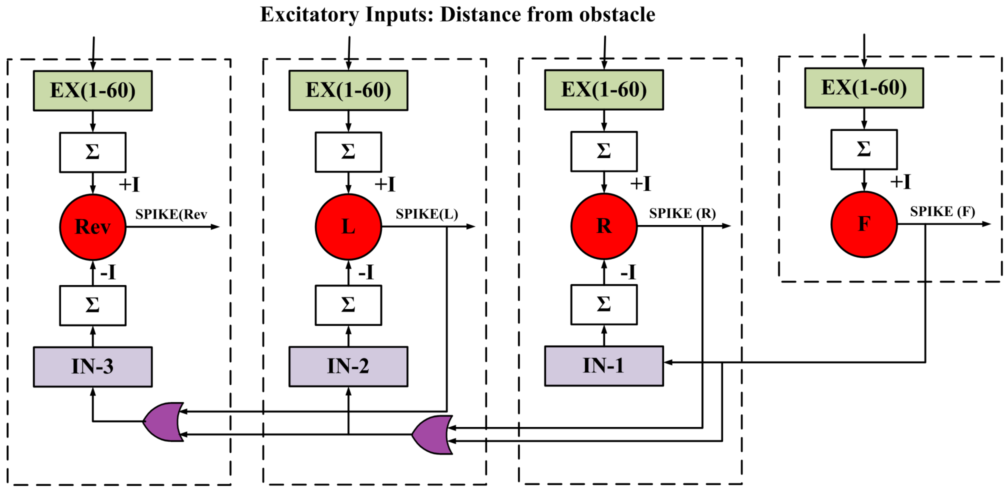

Fig. 4: **Reduced architecture of the robotic controller implemented on the FPGA** The neuron F,R,L and Rev correspond to motor neurons driving motors on forward, right, left and reverse directions of the robotic car respectively. The motors neurons directly inhibit their predecessor neurons. For example neuron F inhibits R, L and Rev neurons. (Tuning blocks are not shown in figure.)

In addition to the motor neuron, the system consists of three inhibitory interneurons (N_1,N_2 and N_3) establishing inhibition in the directions of lower priority. The inhibitory neurons are connected to the respective motor neurons by a set of 40 inhibitory synapses. The inhibitory neurons are connected with their pre-synaptic neuron using excitatory synapses. For example, the motor neuron F forms an excitatory synaptic connection with the inhibitory neuron N_1. N_1 forms inhibitory synaptic interconnections with motor neurons R, L and Rev, thereby inhibiting right, left and reverse movements in the absence of an obstacle in the forward direction. Similarly, motor neuron R inhibit movements in left and reverse directions through interneuron N_2 if there is no obstacle in the right direction. This topology follows all motor neurons in order of priority. In our experiment forward direction has the highest priority and the reverse direction has the lowest priority–($Forward > Right > Left > Reverse$).

B. Reduced Architecture Establishing Obstacle Avoidance Task

In general, neurons use either excitatory or inhibitory neurotransmitter to communicate with their target neuron, i.e. they are either glutamatergic or GABAergic, respectively. As opposed to this observation, some neurons have been found to produce both type of neurotransmitters (excitatory and inhibitory) depending on their target regions or neurons. Some neurotransmitters can cause both excitatory and inhibitory behavior (acetylcholine and dopamine) [43], [44] . For hardware, we try to utilize this fact in reducing the implementation overhead. Based on this observation, although motor neurons are generally excitatory, we combine inhibitory behavior to this thereby avoiding the role of inhibitory interneuron. Thus, in the reduced architecture, the motor neurons directly inhibit the movements in directions of lower priority.

Another important reduction comes from the fact that we design a shared synaptic approach. Dendrites of a neuron receive inputs from multiple neurons [36], [45]. Hence we incorporate this observation using a single synapse which processes all inputs to a neuron. All the inhibitory impulses are obtained through the same set of synapses. For example, the neuron 'Rev' gets inhibitory impulses from 'L', 'R' and 'F' neurons through the same set of synapses (We use a digital OR gate for this purpose).

Finally, the set of 40 inhibitory synapses is replaced by a single large synapse, which injects the same current as would be provided by 40 separate synapses. This reduction gives a more compact design, but a trade off may be reduced fault tolerance. The reduced architecture incorporating these three observations are presented in Figure. 4.

Although the same task is achievable by both the complete and reduced schemes, the presence of higher number of synapses increases the capability of repair. In the reduced scheme, if the single inhibitory synapse is faulty, there is no room for inhibition of movements in undesired directions, and the system will fail. For environments which do not possess serious faults in the system, the proposed scheme of reduced inhibitory synapses can be used. The count of excitatory and inhibitory synapses are selected based on the requirement of the environment in which the robot car is to be employed. Higher the synapses, higher is the reliability to faults, but the cost is in hardware consumption. Hence there is a trade-off between fault-tolerance and hardware overhead. Therefore for more challenging environments such as space, we can include additional synapses with extra hardware area but with the capacity to have a higher tolerance to faults. This suits such cases as it is not possible to replace such systems once deployed. So hardware cost becomes less important and reliability a higher priority.

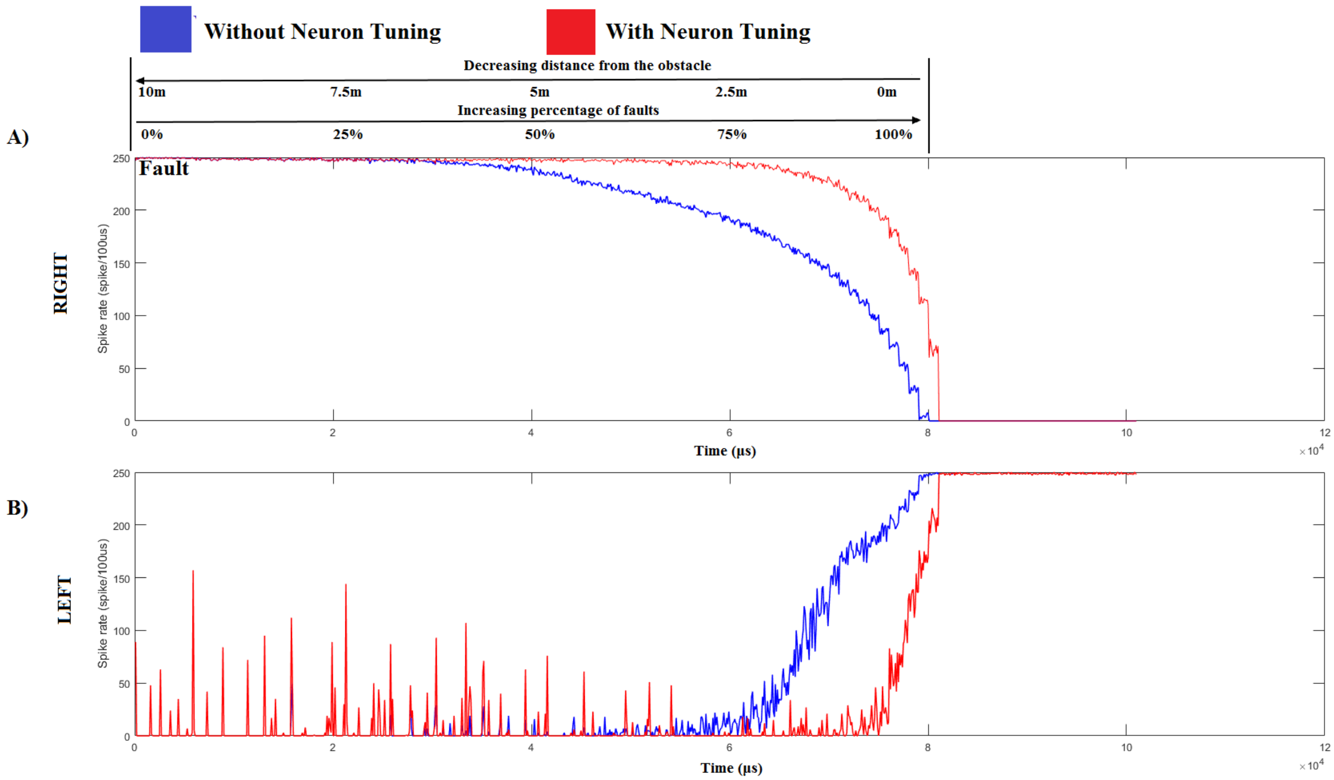

Fig. 5: **Illustration of proposed homeostatic regulation** (A) Performance of Right neuron (B) Performance of Left neuron. Note: Results shown are for threshold voltage based tuning. Similar results are achievable for DPR based tuning.

VI. Experimental Results

The proposed architecture of neuron self-tuning for a robotic controller was implemented in Xilinx ISE 14.7 and simulated using the Xilinx Isim simulator. The system is benchmarked against a fault-free controller implemented on the FPGA. We deliberately induced faults in the system to establish the concept of fault recovery. In presence of faults of various grades, the proposed system could successfully establish fault-free behavior using the proposed neuronal self-tuning concept.

A. Homeostatic Regulation

Figure. 5 shows the homeostatic regulation of firing rate. We consider the case (initial state) in which the Right and Left directions are free of an obstacle. Then, the synapses associated with Right direction are made faulty by introducing faults ranging 0 to 100%. This leads to a reduction in current associated with Right neuron and the current band changes to establish homeostasis. During this experiment, we consider the left direction to be free of an obstacle. The system could achieve a constant rate up to 75% degradation in the synaptic input. Thereafter, the forward neuron spike rate degrades and that of the left neuron improves. We can see that in the presence of neuron tuning (red), in the presence of faults, firing rate is recovering thereby establishing intrinsic homeostasis. In the absence of neuron tuning (blue), the neuron performance degrades at a faster rate. The curve plotted in Red is rapid compared to the one in Blue. This means that the tuned behavior helps a sharper transition from one state to another, while a robot moves in the field. The detection of obstacle happens from a distance but the movement is stopped only when the robot reaches $2.5 meters$ close to the obstacle. (25% close to the obstacle. The range of the robot is considered as $10m$.). The spikes in the right direction are lower than the desired rate of $250 spikes/100\mu s$, when the obstacle is 25% close to the robot (similar to the case when a fault is higher than 75%). Once the firing rate reduces considerably, the left motor neuron starts to fire as shown in Figure. 5-B. The rise in firing rate of the left neuron is also controlled in the presence of neuron tuning. Hence we observe the left neuron firing only if there is a sufficiently large fault in the right neuron.

This is similar to a condition in which the robot approaches an obstacle on the right, whereas no obstacle is present on the left. The right motor neuron should cease to fire and the left motor neuron should increase firing. The presence of neuron self-tuning smoothes out this process.

B. Robotic Controller Implemented on FPGA Achieving Obstacle Avoidance task

The robotic controller would detect the trigger for a particular direction if spike rate is $300 spikes/100\mu s$ for the

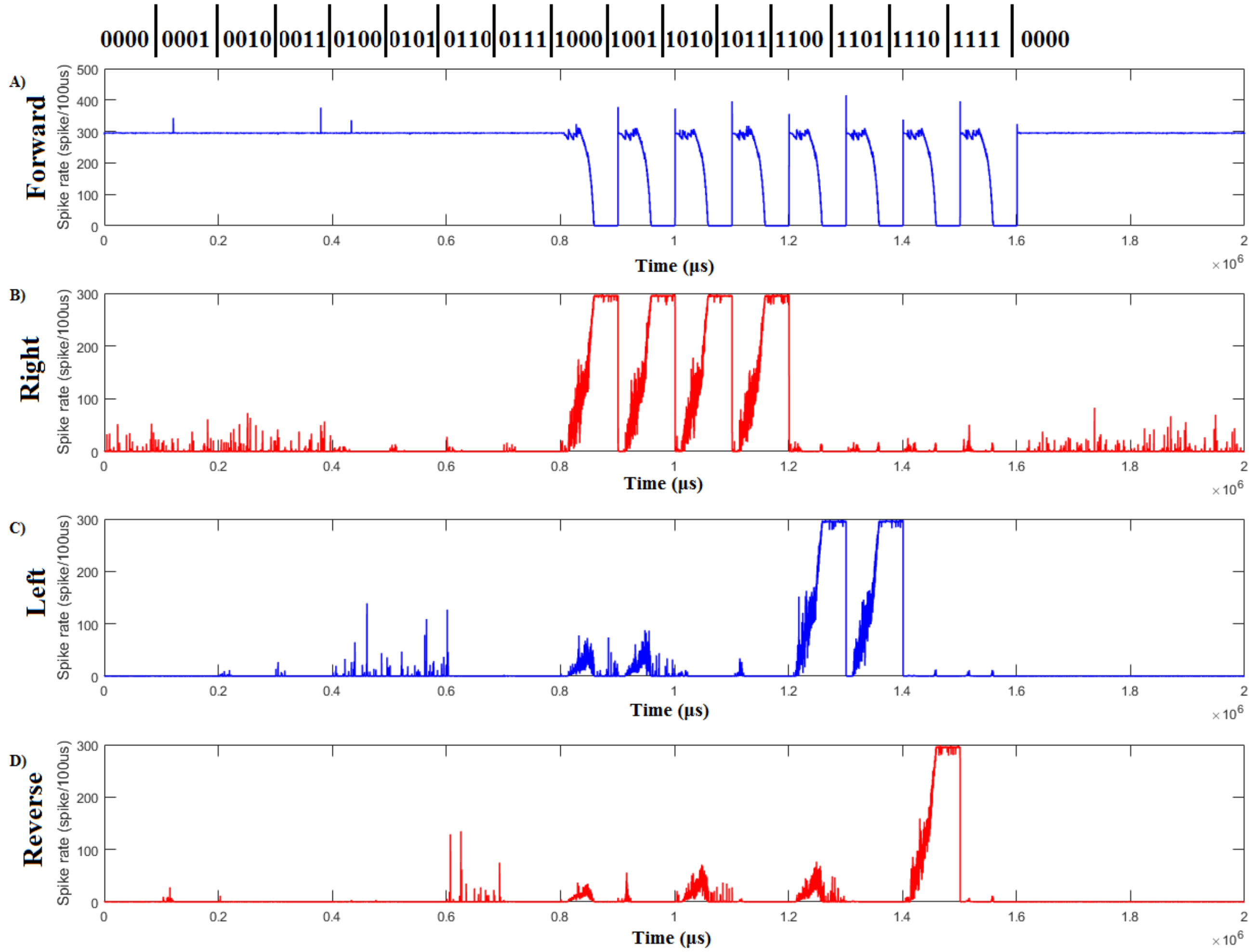

Fig. 6: **Illustration of proposed obstacle avoidance task with neuronal self tuning through DPR of neuron clock** (A) Performance of Forward neuron (B) Performance of Right neuron (C) Performance of Left neuron (D) Performance of Reverse neuron. The vertical line on top of the figure indicate transition between different combinations of obstacles.

DPR based scheme ($250spikes/100\mu s$ for the threshold voltage based scheme). In our experiments, we tested multiple combinations of obstacles in different positions of the robotic field. In Figure. 6, the movements of four motor neurons employing DPR based neuron tuning is projected. The usage of four bits in the paper does not mean that only four bits in representing the obstacle presence/absence in the FPGA implementation. The presentation of '1000' in the paper means the condition in which the robot started from no obstacle to a state of an obstacle in close proximity in the forward direction. All the 16 states depicted in Figure. 6 shows the position of robot movement robot movement from a state of no obstacle in all directions to the state shown in the figure. That is '0000' means 0 (60 bit long) applied to synapses of the Forward, Right, Left and Reverse neuron. This indicates the field is free of an obstacle for $10m$ in all directions. The presentation of '1000' means that all the forward synapses have applied a logic one. This is a short representation of the final state of the robot synapses. The first one in'1000' indicates a transition from 0 to 'all ones' for synapses associated with the Forward neuron, through all intermediate transitions. The remaining zeros indicate no transition(0)in Right, Left or Reverse direction. Hence the spikes on motor neuron 'R' increases leading to a right movement. The encoding '1110' corresponds to a movement from a field of no obstacle to a close encounter of an obstacle on the forward, right and left direction, leading to a reverse movement in the robot car.

The spike outputs from the motor neurons are used to control the motors of the robot. We observe that the frequency of the neuron spikes is not fixed and has fluctuations. A moving mean algorithm can be used to smooth out the short-term frequency fluctuations and highlight longer trends [8]. This is given by

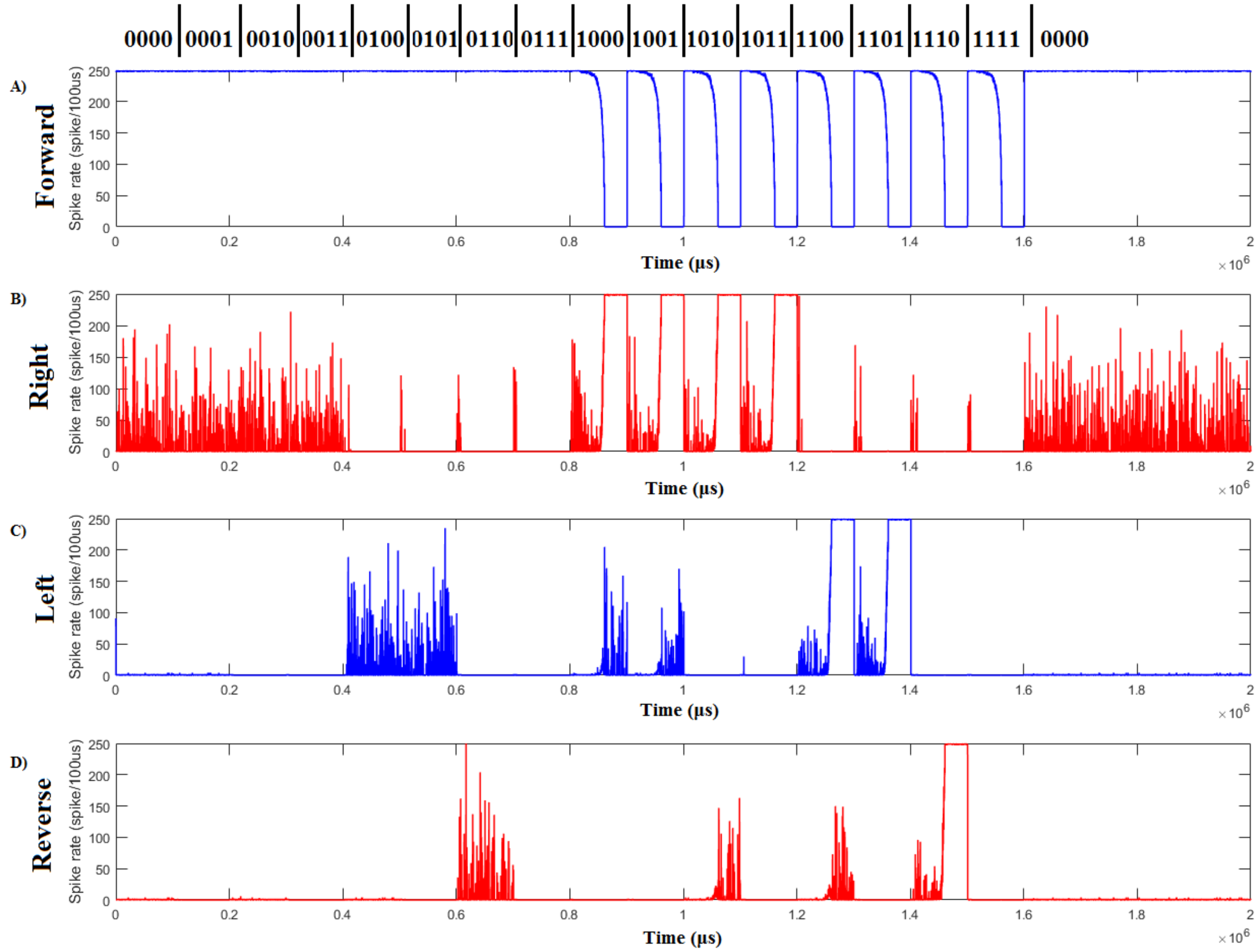

Fig. 7: **Illustration of proposed obstacle avoidance task with neuronal self tuning through threshold voltage adjustment of neuron** (A) Performance of forward neuron (B) Performance of right neuron (C) Performance of left neuron (D) Performance of reverse neuron. The vertical line on top of the figure indicate transition between different combinations of obstacles.

Eq. (5).

$$f' = \frac{1}{n}\sum_{i=0}^{n-1} f_{m-i} \tag{5}$$

where f' is the average frequency which is used to control the robot motor, m is the current time point, n is the size of the neuron frequency subset, and f_{m-i} is the neuron frequency at the time point of $(m-i)$.

Figure. 7 depicts the movements of four motor neurons employing threshold voltage based neuron tuning. The plots have a similar meaning as in Figure. 6. The plots in Figure.6 and Figure. 7 represent the raw spike data before averaging Eq. (5) is applied. The averaging would further smooth out the process. The motor directions are chosen based on majority decision. We can see that the fluctuations while establishing homeostasis are lesser in threshold voltage based neuron tuning than the DPR based scheme, proving to be a more efficient strategy. The DPR based scheme is designed to establish homeostasis (on the dominating neuron) by delivering spikes with a rate of $300 spikes/100\mu S$. Any glitches occurring while establishing this rate is treated as ‘fluctuations while establishing homeostasis’. For the threshold voltage based scheme, the homeostasis is achieved on the dominating direction when the spike rate is $250 spikes/100\mu S$. Any glitches occurring while establishing this rate for the threshold voltage based scheme is referred to as ‘fluctuations while establishing homeostasis’. The spike rate for achieving homeostasis are selected randomly. On comparing Figure. 6 and Figure. 7, We could see more fluctuations on the dominating neurons in Figure. 6 than Figure. 7. (For case 0111, the dominating neuron is Forward neuron which has some fluctuations in Figure. 6 but is smooth in Figure. 7).

The spikes present (Figure. 7) for the condition ‘0000’ for

TABLE III: **Statistical Comparison of Firing Activity of Complete and Reduced Architecture**

Test Vector F-R-L-Rev	Test Direction	Average Firing activity Complete Model (spikes/μs)	Average Firing activity Reduced Model (spikes/μs)
0000	$Forward$	2.50	2.50
0001	$Forward$	2.48	2.48
0010	$Forward$	2.49	2.49
0011	$Forward$	2.49	2.49
0100	$Forward$	2.49	2.49
0101	$Forward$	2.49	2.49
0110	$Forward$	2.48	2.48
0111	$Forward$	2.48	2.48
1000	$Right$	2.49	2.48
1001	$Right$	2.49	2.49
1010	$Right$	2.49	2.49
1011	$Right$	2.49	2.49
1100	$Left$	2.49	2.49
1101	$Left$	2.48	2.48
1110	$Reverse$	2.49	2.49
Note: Test vector F-R-L-Rev=1010 indicate that there is obstacle in forward and left directions and the path is clear for 10 meters in directions right and reverse. This indicate that the robot has to take movement in the right direction. Note: Results shown are for threshold voltage based tuning. Similar results are achievable for DPR based tuning.			

the Right neuron, does not mean the tendency to lean towards the right. These spikes are sparse and hence these stochastic behavior are insufficient to control the movements to the Right. We see some unwanted spikes in directions of less priority. The glitches are not completely removed in our work in order to encourage spontaneous activity in the network. The glitches in the undesired directions can be completely eliminated by increasing inhibitory synapse count. But, this is not a necessary requirement due to the fact listed in the above statement. In our application, we have used 40 synapses as inhibitory.

As described in Section IV, the transition from one stage to another is decided based on the maximum current injected to the neuron. In our experiment, this is performed in $2\mu s$. Once the change in the band is detected, for threshold voltage adjustment scheme, the threshold voltage is updated in one single clock cycle. For DPR based scheme, a small duration (order of μs) is required before switching to the higher desired band frequency. Hence in both the schemes, the neuron tuning happens in μs, making the system highly responsive in detecting the changes in the environment due to obstacles. In real world applications, the obstacles approaching the robot at this faster rate is quite unusual.

C. Statistical Comparison of Complete and Reduced Architecture

In our experiments, we tried multiple combinations of obstacles in different positions of the robotic field ranging from the encoding '0000' to '1111' for the complete model and the reduced model of robotic controller. A statistical test was performed on the spike trains generated by the two models subjected to the same input field condition. Table III shows the result of average spiking activity in the four directions of movement. We found the percentage change in average firing activity of the two models subjected to the same test condition. The results are close to the '0' indicating the samples to be statistically indistinguishable.

TABLE IV: **Hardware Overhead of the complete robotic controller architecture**

Methodology/Components	Slice	Slice Reg	LUT	PLL/MMCM	DSP
V_{th} based tuning unit/neuron	3	5	3	0	0
DPR based tuning unit/neuron	17	22	28	1	0
V_{th} based complete design	5620	9821	7499	1	28
V_{th} based reduced design	3770	8756	3789	1	16
DPR based reduced design	3798	8827	3852	5	16

TABLE V: **Total on-chip power and maximum operating frequency**

Methodology	On-chip power (W)	Maximum operating frequency (MHz)
V_{th} based complete design	1.252	16.199
V_{th} based reduced design	1.240	25.515
DPR based reduced design	1.513	30.913

D. Hardware Results on Xilinx Artix-7 FPGA

The proposed methodology is implemented on the Xilinx Artix-7 FPGA board. Recovery of firing rates in the proposed methodology, implemented on the FPGA is monitored using the *Xilinx ChipScope Pro* analyzer. Power estimation of the circuits was carried out using *Xilinx XPower Analyzer* and delay estimation using *Xilinx Timing Analyzer*. Table IV reports the hardware resource footprint of the proposed models. Estimated total on-chip power dissipation and the maximum operating frequency of the overall proposed architectures is shown in Table V. As evident from these reports, the proposed neuronal tunability for homeostatic regulation of firing rate of the robotic controller can be implemented with reduced hardware overhead and power consumption.

VII. Conclusion

In this paper, we discussed three novelties. Firstly, we built a homeostatic fault tolerant bio-inspired architecture for spiking neural networks. We explored two methods for incorporating fault tolerant homeostasis into a system. The proposed system is adaptive to the changes in the system and does not require dedicated units for detection and correction. Even if some of the inputs are faulty, the system could establish the task by enhancing the neuron features using the signals delivered by the fault free interconnects. Secondly, the proposed idea is demonstrated on an FPGA mapped with a robotic controller application. Finally, we created a reduced model of the architecture establishing the same task. The robot would move smoothly if the neuron inputs are at least 25% of the normal signal. This is incorporated to avoid unusual behavior while establishing movement in any direction and would take care of noise equivalent to 75% reduction in synaptic input. Hence, the system would work over a broad range of input values. But the presence of obstacle and noise (together) would definitely fasten the transition and requires much more research effort in investigating this complex behavior. We may require improvements in the above architecture if the noise levels are extremely high. We are currently working on real robot implementation of FPGA based SNN which could analyze the presence of noisy sensor data and is treated as a future work. Inspirations derived from different biological processes are incorporated in building this architecture. The proposed design is appropriate for FPGA-based applications running in environments that induce faults in systems, where reliability is a critical factor. The work presented represents an initial step towards a new form of fault tolerant designs, with low overhead and high performance.

VIII. Acknowledgements

The work is part of the SPANNER project and is funded by EPSRC grant–(EP/N007050/1, EP/N00714X/1). Additionally, the authors would like to acknowledge the platform grant–(EP/K040820/1) funded by EPSRC.

References

[1] X.-S. Yang, Z. Cui, R. Xiao, A. H. Gandomi, and M. Karamanoglu, *Swarm Intelligence and Bio-inspired Computation: Theory and Applications*. Newnes, 2013.

[2] M. Opendak and E. Gould, "Adult Neurogenesis: a Substrate for Experience-Dependent Change," *Trends in cognitive sciences*, vol. 19, no. 3, pp. 151–161, Feb. 2015.

[3] G. W. Davis and I. Bezprozvanny, "Maintaining the Stability of Neural Function: a Homeostatic Hypothesis," *Annual Review of Physiology*, vol. 63, no. 1, pp. 847–869, 2001.

[4] I. van Welie, J. A. van Hooft, and W. J. Wadman, "Homeostatic Scaling of Neuronal Excitability by Synaptic Modulation of Somatic Hyperpolarization-Activated Ih Channels," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 101, no. 14, pp. 5123–5128, 2004.

[5] T. C. Südhof and R. C. Malenka, "Understanding Synapses: Past, Present, and Future," *Neuron*, vol. 60, no. 3, pp. 469–476, 2008.

[6] M. Pecht and V. Ramappan, "Are Components Still the Major Problem: a Review of Electronic System and Device Field Failure Returns," *IEEE Transactions on Components, Hybrids, and Manufacturing Technology*, vol. 15, no. 6, pp. 1160–1164, 1992.

[7] R. D. Schrimpf and D. M. Fleetwood, *Radiation Effects and Soft Errors in Integrated Circuits and Electronic Devices*. World Scientific, 2004, vol. 34.

[8] J. Liu, J. Harkin, L. McDaid, D. M. Halliday, A. M. Tyrrell, and J. Timmis, "Self-repairing mobile robotic car using astrocyte-neuron networks," in *2016 International Joint Conference on Neural Networks (IJCNN)*, Jul. 2016, pp. 1379–1386.

[9] J. Liu, J. Harkin, L. P. Maguire, L. J. McDaid, J. J. Wade, and G. Martin, "Scalable Networks-on-Chip Interconnected Architecture for Astrocyte-Neuron Networks," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 63, no. 12, pp. 2290–2303, Dec. 2016.

[10] A. P. Johnson, D. M. Halliday, A. G. Millard, A. M. Tyrrell, J. Timmis, J. Liu, J. Harkin, L. McDaid, and S. Karim, "An FPGA-based Hardware-Efficient Fault-Tolerant Astrocyte-Neuron Network," in *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, Dec. 2016, pp. 1–8.

[11] Q. Wang, Y. Li, and P. Li, "Liquid state machine based pattern recognition on FPGA with firing-activity dependent power gating and approximate computing," in *2016 IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2016, pp. 361–364.

[12] Q. Wang, Y. Li, B. Shao, S. Dey, and P. Li, "Energy efficient parallel neuromorphic architectures with approximate arithmetic on FPGA," *Neurocomputing*, vol. 221, pp. 146–58, Jan. 2017.

[13] S. Merchant, G. D. Peterson, S. K. Park, and S. G. Kong, "FPGA Implementation of Evolvable Block-based Neural Networks," in *2006 IEEE International Conference on Evolutionary Computation*, Jul. 2006, pp. 3129–3136.

[14] N. Jing, J.-Y. Lee, Z. Feng, W. He, Z. Mao, and L. He, "SEU Fault Evaluation and Characteristics for SRAM-based FPGA Architectures and Synthesis Algorithms," *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 18, no. 1, p. 13, Jan. 2013.

[15] Q. Xia, W. Robinett, M. W. Cumbie, N. Banerjee, T. J. Cardinali, J. J. Yang, W. Wu, X. Li, W. M. Tong, D. B. Strukov *et al.*, "Memristor-CMOS Hybrid Integrated Circuits for Reconfigurable Logic," *Nano letters*, vol. 9, no. 10, pp. 3640–3645, 2009.

[16] X. Liu, Z. Zeng, and S. Wen, "Implementation of Memristive Neural Network With Full-Function Pavlov Associative Memory," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 63, no. 9, pp. 1454–1463, Sep. 2016.

[17] S. P. Adhikari, H. Kim, R. K. Budhathoki, C. Yang, and L. O. Chua, "A Circuit-Based Learning Architecture for Multilayer Neural Networks With Memristor Bridge Synapses," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 62, no. 1, pp. 215–223, Jan. 2015.

[18] J. A. Starzyk and Basawaraj, "Memristor Crossbar Architecture for Synchronous Neural Networks," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 61, no. 8, pp. 2390–2401, Aug. 2014.

[19] A. Vatankhahghadim, W. Song, and A. Sheikholeslami, "A Variation-Tolerant MRAM-Backed-SRAM Cell for a Nonvolatile Dynamically Reconfigurable FPGA," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 62, no. 6, pp. 573–577, Jun. 2015.

[20] J. J. Davis and P. Y. Cheung, "Achieving low-overhead fault tolerance for parallel accelerators with dynamic partial reconfiguration," in *24th International Conference onField Programmable Logic and Applications (FPL)* . IEEE, Sep. 2014, pp. 1–6.

[21] A. Upegui, J. Izui, and G. Curchod, "Fault mitigation by means of dynamic partial reconfiguration of Virtex-5 FPGAs," in *International Conference on Reconfigurable Computing and FPGAs (ReConFig)*. IEEE, Dec. 2012, pp. 1–6.

[22] A. L. Hodgkin and A. F. Huxley, "Propagation of Electrical Signals Along Giant Nerve Fibres," *Proceedings of the Royal Society B: Biological Sciences*, pp. 177–183, 1952.

[23] N. Fourcaud-Trocmé, D. Hansel, C. V. Vreeswijk, and N. Brunel, "How Spike Generation Mechanisms Determine the Neuronal Response to Fluctuating Inputs," *Journal of Neuroscience*, vol. 23, no. 37, pp. 11 628–11 640, 2003.

[24] E. M. Izhikevich, "Simple Model of Spiking Neurons," *IEEE Transactions on Neural Networks*, vol. 14, no. 6, pp. 1569–1572, 2003.

[25] E. M. Izhikevich and R. FitzHugh, "FitzHugh-Nagumo Model," *Scholarpedia*, vol. 1, no. 9, p. 1349, 2006.

[26] J. L. Hindmarsh and R. Rose, "A Model of Neuronal Bursting Using Three Coupled First Order Differential Equations," *Proceedings of the Royal Society of London B: Biological Sciences*, vol. 221, no. 1222, pp. 87–102, 1984.

[27] N. Brunel and M. C. van Rossum, "Quantitative Investigations of Electrical Nerve Excitation Treated as Polarization," *Biological Cybernetics*, vol. 97, no. 5, pp. 341–349, 2007.

[28] W. Gerstner and W. M. Kistler, *Spiking Neuron Models: Single Neurons, Populations, Plasticity*. Cambridge university press, 2002.

[29] E. Orhan, "The Leaky Integrate-and-Fire Neuron Model," [Online]. Available: http://www.cns.nyu.edu/ eorhan/notes/lif-neuron.pdf, Nov. 2012.

[30] J. L. Franklin, D. J. Fickbohm, and A. Willard, "Long-term Regulation of Neuronal Calcium Currents by Prolonged Changes of Membrane Potential," *Journal of Neuroscience*, vol. 12, no. 5, pp. 1726–1735, 1992.

[31] T. O'Leary, M. C. van Rossum, and D. J. Wyllie, "Homeostasis of Intrinsic Excitability in Hippocampal Neurones: Dynamics and Mechanism

of the Response to Chronic Depolarization," *The Journal of physiology*, vol. 588, no. 1, pp. 157–170, 2010.

[32] J. Breder, C. F. Sabelhaus, T. Opitz, K. G. Reymann, and U. H. Schröder, "Inhibition of Different Pathways Influencing Na+ Homeostasis Protects Organotypic Hippocampal Slice Cultures from Hypoxic/Hypoglycemic Injury," *Neuropharmacology*, vol. 39, no. 10, pp. 1779–1787, 2000.

[33] P. A. Sotiropoulou and C. Blanpain, "Development and Homeostasis of the Skin Epidermis," *Cold Spring Harbor perspectives in biology*, vol. 4, no. 7, Jul. 2012.

[34] Z. Liu, J. Golowasch, E. Marder, and L. Abbott, "A Model Neuron with Activity-dependent Conductances Regulated by Multiple Calcium Sensors," *Journal of Neuroscience*, vol. 18, no. 7, pp. 2309–2320, 1998.

[35] H. Hultborn, S. Lindström, and H. Wigström, "On the Function of Recurrent Inhibition in the Spinal Cord," *Experimental Brain Research*, vol. 37, no. 2, pp. 399–403, 1979.

[36] G. M. Shepherd, *The Synaptic Organization of the Brain*. Oxford University Press, 2003.

[37] R. Bonamy, S. Bilavarn, D. Chillet, and O. Sentieys, "Power consumption models for the use of dynamic and partial reconfiguration," vol. 38, no. 8, Nov. 2014, pp. 860–872.

[38] *7 Series FPGAs Clocking Resources UG472 (v1.12)*, Xilinx Inc, [Online]. Available: https://www.xilinx.com/support/documentation/user_guides/ug472_7Series_Clocking.pdf, Sep. 2016, accessed: 2017-02-03.

[39] *Xilinx 7 Series FPGA and Zynq-7000 All Programmable SoC Libraries Guide for HDL Designs UG768 (v14.7)*, Xilinx Inc, [Online]. Available: https://www.xilinx.com/support/documentation/sw_manuals/xilinx14_7/7series_hdl.pdf, Oct. 2013, accessed: 2017-02-03.

[40] K. Paulsson, M. Hübner, S. Bayar, and J. Becker, "Exploitation of Runtime Partial Reconfiguration for Dynamic Power Management in Xilinx Spartan III-based Systems," in *In Proceedings of International Workshop on Reconfigurable Communication-centric Systems-on-Chip*, 2007.

[41] K. He, L. Crockett, and R. Stewart, "Dynamic reconfiguration technologies based on FPGA in software defined radio system," *Journal of Signal Processing Systems*, vol. 69, no. 1, pp. 75–85, Oct. 2012.

[42] Anju P. Johnson and Rajat Subhra Chakraborty and Debdeep Mukhopadyay, "An Improved DCM-Based Tunable True Random Number Generator for Xilinx FPGA," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 64, no. 4, pp. 452–456, Apr. 2017.

[43] B. Kolb, I. Q. Whishaw, U. Hasson, and Y. Shavit, *An Introduction to Brain and Behavior*. Worth New York, 2001.

[44] R. J. Romanelli, J. T. Williams, and K. A. Neve, "Dopamine Receptor Signaling: Intracellular Pathways to Behavior," in *The Dopamine Receptors*. Springer, Sep. 2009, pp. 137–173.

[45] A. W. Moore, L. Y. Jan, and Y. N. Jan, "Hamlet, a Binary Genetic Switch Between Single- and Multiple- Dendrite Neuron Morphology," *Science*, vol. 297, no. 5585, pp. 1355–1358, 2002.

Anju P. Johnson Anju P. Johnson is a Postdoctoral Research Associate in the Department of Electronic Engineering of University of York, U.K. since 2016. She received B.Tech degree in Electronics and Communication Engineering from Cochin University of Science and Technology (CUSAT) in the year 2010, M.Tech degree in VLSI Design from Amrita University in 2012 and Ph.D. in Computer Science and Engineering from Indian Institute of Technology (IIT) Kharagpur in the year 2016. Before Joining University of York, she was a Senior Project Officer in the Department of Computer Science and Engineering of IIT Kharagpur (2012-2016) and a faculty (Adhoc) in the Department of Electronics and Communication Engineering, National Institute of Technology (NIT) Calicut (2012). Her research interests include Internet of Things, Hardware Security, Neuromorphic Computing and FPGA Prototyping.

Junxiu Liu Junxiu Liu received the PhD degree from University of Ulster, UK, with the support of a Vice-Chancellors Research Scholarship. His research interests relate to: neural glial system, and its implementations and applications.

Alan G. Millard Alan G. Millard received his Ph.D. degree in Computer Science, with a focus on fault detection in swarm robotic systems, from the University of York in 2017. He is currently a research associate in the Department of Electronic Engineering at the University of York, working on the SPANNER project. His current research focus is on engineering fault tolerant swarm robotic systems.

Shvan Karim Shvan Karim holds a B.Sc. degree in Electrical Engineering from the University of Sulaimani, Kurdistan, Iraq, a M.Sc. degree from the University of Southampton, UK, and is currently a PhD candidate at Ulster University, UK.

Andy M. Tyrrell Prof. Andy M. Tyrrell received a 1st class honours degree in 1982 and a PhD in 1985 (Aston University), both in Electrical and Electronic Engineering. He joined the Department of Electronic Engineering at the University of York in April 1990, he was promoted to the Chair of Digital Electronics in 1998. Previous to that he was a Senior Lecturer at Coventry Polytechnic. Between August 1987 and August 1988 he was visiting research fellow at Ecole Polytechnic Lausanne Switzerland, where he was researching into the evaluation and performance of multiprocessor systems. From September 1973 to September 1979 he worked for STC at Paignton Devon, on the design and development of high frequency devices. He is a Senior member of the IEEE and a Fellow of the IET.

Jim Harkin Jim Harkin received the B.Tech. degree in electronic engineering, the M.Sc. degree in electronics and signal processing, and the Ph.D. degree from Ulster University, Coleraine, U.K., in 1996, 1997, and 2001, respectively. He is currently a Reader with the School of Computing and Intelligent Systems, Ulster University, Derry, U.K. He has authored over 80 articles in peer-reviewed journals and conferences. His current research interests include the design of intelligent embedded systems to support self-repairing capabilities and the hardware/software implementation of spiking neural networks.

Jon Timmis Jon Timmis received the Ph.D. degree in computer science from the University of Wales, Aberystwyth, in 2000, where he also worked as a Research Associate investigating the use of immune system metaphors for machine learning. He is a Professor and Head of Electronic Engineering Department at the University of York, York, U.K. His primary research interest is in the computational abilities of the immune system and how they relate to computer science and engineering. From 2000 to 2005, he was a Lecturer, then Senior Lecturer at the Computing Laboratory, University of Kent, where he was head of the Applied and Interdisciplinary Informatics (AII) Research Group. Dr. Timmis, in collaboration with Dr. P. Bentley, founded the International Conference on Artificial Immune Systems (ICARIS) in 2002, and is now Chair of the ICARIS Steering Committee. He is on the Editorial Board of Evolutionary Computation (MIT Press).

Liam J. McDaid Liam J. McDaid received the B.Eng. (Hons.) degree in Electrical and Electronics Engineering and the Ph.D. degree in solid-state devices from the University of Liverpool, Liverpool, U.K., in 1985 and 1989, respectively. He is currently a Professor of Computational Neuroscience with Ulster University, Coleraine, U.K., and leads the Computational Neuroscience and Neural Engineering Research Team. He has coauthored over 120 publications in his career to date. His current research interests include modeling the role of glial cells in the functional and dysfunctional brain and he is also involved in the development of software/hardware models of neural-based computational systems, with particular emphasis on the mechanisms that underpin self-repair in the human brain. Dr. McDaid received several research grants in this domain and is currently a Collaborator on an HFSP and EPSRC funded project.

David M. Halliday David M. Halliday is a Reader in the Department of Electronic Engineering of University of York the University of York, York, U.K. He received the B.Sc. and Ph.D. degrees in electronics and electrical engineering from the University of Glasgow, Scotland, U.K. He has held research positions at the University of Glasgow, University of Strathclyde, and NTT Basic Research Laboratories, Tokyo. His research interests are in the interdisciplinary fields of Computational Neuroscience and Neural Computing.